Theoretical Computer Science 412 (2011) 5121–5131

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

# Randomized competitive algorithms for online buffer management in the adaptive adversary model☆

Marcin Bienkowski [a], Marek Chrobak [b], Łukasz Jeż [a,*]

[a] Institute of Computer Science, University of Wrocław ul. Joliot-Curie 15, 50-383 Wrocław, Poland
[b] University of California, Riverside, CA 92521, USA

ARTICLE INFO

*Article history:*
Received 6 July 2010
Received in revised form 20 October 2010
Accepted 9 May 2011
Communicated by T. Erlebach

*Keywords:*
Packet scheduling
Online scheduling
Buffer management
Competitive analysis
Adaptive adversary

ABSTRACT

In the problem of buffer management with bounded delay, packets with weights and deadlines arrive at a network switch over time, and the goal is to send those packets on the outgoing link while maximizing the total weight of the packets that are sent before their deadlines. We present a study of randomized algorithms that are competitive against an adaptive adversary. Previous studies considered only the oblivious adversary model that does not capture dependency of network traffic on the packet scheduling algorithm. We give a new analysis of a previously known algorithm, which shows that it remains $e/(e-1)$-competitive even against an adaptive adversary. We complement this with a 4/3 lower bound on the competitive ratio on 2-bounded instances, in which each packet has a lifespan of one or two steps. We also study more restricted 2-uniform instances, in which every packet has a lifespan of exactly two steps. For such instances we give a 1.2 lower bound on the competitive ratio of arbitrary algorithms and 4/3 lower bound on the competitive ratio of memoryless scale-invariant algorithms. Finally, we devise a 4/3-competitive memoryless scale-invariant algorithm for 2-bounded instances, matching two of these lower bounds.



## 1. Introduction

In this paper, we consider the problem of *buffer management with bounded delay*, introduced by Kesselman et al. [1]. This problem models the behavior of a single network switch responsible for scheduling packet transmissions along an outgoing link. We assume that time is divided into unit-length steps. At the beginning of a time step, any number of packets may arrive at a switch and be stored in its *buffer*. Each packet has a positive weight and a deadline, which specifies the latest time when the packet can be transmitted. Only one packet from the buffer can be transmitted in a single step; it is removed from the buffer upon transmission. The goal is to maximize the *gain*, defined as the total weight of the transmitted packets.

We note that *buffer management with bounded delay* is equivalent to a scheduling problem in which packets are represented as jobs of unit length, with given release times, deadlines and weights. Release times and deadlines are restricted to integer values. In this setting, the goal is to maximize the total weight of jobs which are completed before their deadlines.

As the process of managing a packet queue is inherently a real-time task, we model it as an online problem. This means that the algorithm, when deciding which packets to transmit, has to base its decision solely on the packets which have already arrived at a switch, without the knowledge of the future. With this in mind, from now on we use the notion of relative deadlines rather than the absolute ones. At any given time step, the *relative deadline* of a packet is simply its absolute

☆ A preliminary version of this paper appeared in the Proc. of the 6th International Workshop on Approximation and Online Algorithms (WAOA 2008).

* Corresponding author. Tel.: +48 71 375 78 29; fax: +48 71 375 78 01.
*E-mail address:* lje@cs.uni.wroc.pl (Ł. Jeż).

0304-3975/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.tcs.2011.05.015

deadline minus the current time. When a packet is injected into the buffer, its relative deadline is initialized, and, while the packet is not chosen for transmission, this relative deadline is decremented by one after every step. As soon as its value reaches zero, the packet expires, and it is consequently removed from the buffer.

*Competitive analysis.* To measure the performance of an online algorithm, we use the standard notion of competitive analysis [2], which, roughly speaking, compares the gain of the algorithm to the gain of the optimal solution on the same instance. For any algorithm ALG, we denote its gain on instance $I$ by $\mathcal{G}_{\text{ALG}}(I)$. The optimal offline algorithm is denoted by OPT. We say that a deterministic algorithm ALG is $\mathcal{R}$-competitive if on any instance $I$ it holds that $\mathcal{G}_{\text{ALG}}(I) \geq \frac{1}{\mathcal{R}} \cdot \mathcal{G}_{\text{OPT}}(I)$.

When analyzing the performance of an online algorithm ALG, we view the process as a game between ALG and an *adversary*. The adversary controls what packets are injected into the buffer and chooses which of them to send. The goal is then to show that the adversary's gain is at most $\mathcal{R}$ times ALG's gain.

If the algorithm is randomized, we consider its expected gain, $\mathbf{E}[\mathcal{G}_{\text{ALG}}(I)]$, where the expectation is taken over all possible random choices made by ALG. However, in the randomized case, the power of the adversary has to be further specified. Following Ben-David et al. [3], we distinguish between an *oblivious* and an *adaptive-online* adversary, which from now on we will call *adaptive*, for short. An oblivious adversary has to construct the whole instance in advance. This instance may depend on ALG but not on the random bits used by ALG during the computation. The expected gain of ALG is compared to the gain of the optimal offline solution on $I$. In contrast, in case of an adaptive adversary, the choice of packets to be injected into the buffer may depend on the algorithm's behavior up to the given time step. This adversary must also provide an answering entity ADV, which creates a solution in parallel to ALG. This solution may not be changed afterward. We say that ALG is $\mathcal{R}$-competitive against an adaptive adversary if for any adaptively created instance $I$ and any answering algorithm ADV, it holds that $\mathbf{E}[\mathcal{G}_{\text{ALG}}(I)] \geq \frac{1}{\mathcal{R}} \cdot \mathbf{E}[\mathcal{G}_{\text{ADV}}(I)]$. We note that ADV is deterministic, but as ALG is randomized, so is the instance $I$.

Observe the following phenomenon: as the instance $I$ is randomized, the adaptive adversary cannot know for sure what packets it will transmit in the future. Consequently, deprived of that knowledge, he cannot ensure any specific order of packet transmissions. Assuming an earliest-deadline order of transmissions in the adversary's schedule is crucial in analyses of the oblivious adversary model. Our analysis in Section 3 does not require any assumption on packet ordering.

In the literature on online algorithms (see e.g. [2]), the definition of the competitive ratio sometimes allows an additive constant, i.e., a deterministic algorithm is then called $\mathcal{R}$-competitive if there exists a constant $\alpha \geq 0$ such that for any instance $I$ it holds that $\mathcal{G}_{\text{ALG}}(I) \geq \frac{1}{\mathcal{R}} \cdot \mathcal{G}_{\text{OPT}}(I) - \alpha$. An analogous definition applies to the randomized case. The bounds we give in this paper are the strongest possible, i.e., our upper bounds hold for $\alpha = 0$, while our lower bounds hold for any constant $\alpha$.

*Bounded sequences.* A packet's *lifespan* is the difference between its deadline and release time. At the time when the packet is injected, its lifespan is thus equal to its relative deadline. We will consider instances with restrictions on packet lifespans. In an *s-bounded instance* the lifespan of each packet is at most $s$, whereas in an *s-uniform instance* the lifespan of each packet is exactly $s$.

### 1.1. Related work

The currently best, 1.828-competitive deterministic algorithm for general instances was given by Englert and Westermann [4]. The lower bound of $\phi \approx 1.618$ for the competitive ratio of deterministic algorithms was shown in [5–7]. The instances which incur this lower bound are in fact 2-bounded. On the other hand, for 2-bounded and 3-bounded instances, deterministic $\phi$-competitive algorithms are known (see [1] and [8], respectively). If we restrict the set of inputs to 2-uniform instances, the optimum competitive ratio equals approximately 1.377 [9].

So far, randomized algorithms have only been studied in the oblivious adversary model. The best known randomized solution is the 1.582-competitive algorithm RMIX by Chin et al. [8]. The best lower bound of 1.25 for randomized algorithms was given by Chin and Fung [6]. This lower bound uses 2-bounded instances, for which a matching upper bound was given by Chin et al. [8]. For 2-uniform instances, the best known lower bound is approximately 1.172 [8]. All currently known bounds are summarized in Table 1.

Most of known algorithms are *memoryless scale-invariant* (refer to Section 1.3 for definition) — but there are a few exceptions. The first one is the best known deterministic algorithm [4] whose competitive ratio is at most 1.828. However, in the very same paper, Englert and Westermann give a 1.893-competitive variant of their algorithm that is memoryless scale-invariant. The optimal deterministic algorithm for 2-uniform instances is another example [9]. In fact, in case of deterministic algorithms for 2-uniform instances, the optimum competitive ratio of memoryless scale-invariant algorithms is 1.414 [10,8], while for non-restricted algorithms the optimal ratio is 1.377 [9].

For a general overview of techniques and results on the buffer management problem, see, e.g., the surveys by Azar [11], Epstein and Van Stee [12] and Goldwasser [13].

### 1.2. Our contribution

In our paper we consider randomized algorithms against an adaptive adversary. In reality, traffic through a switch is not at all independent of the packet scheduling algorithm. For example, lost packets are typically resent, and low throughput

**Table 1**
Comparison of known and new results. The results of this paper are shown in boldface. The results without citations are implied by other entries of the table. An asterisk denotes that the bound is restricted to memoryless scale-invariant algorithms.

| | | Deterministic | (Rand.) adaptive | | (Rand.) oblivious |
|---|---|---|---|---|---|
| General input | Upper | 1.828 [4], 1.893* [4] | 1.828, 1.893* | **1.582** | 1.582* [8] |
| | Lower | 1.618 | 1.25 | **1.333** | 1.25 |
| 2-bounded | Upper | 1.618* [1] | 1.618* | **1.333*** | 1.25* [8] |
| | Lower | 1.618 [5–7] | 1.25 | **1.333** | 1.25 [6] |
| 2-uniform | Upper | 1.377 [9], 1.414* [10] | 1.377, 1.414* | **1.333*** | 1.25* |
| | Lower | 1.377 [9], 1.414* [8] | 1.172 | **1.2, 1.333*** | 1.172 [8] |

through a node can affect the choice of routes for data streams in a network. These phenomena can be captured by the adaptive adversary model but not by the oblivious one. The adaptive adversary model is also of its own theoretical interest and it has been studied in numerous other settings, see [2].

To our knowledge, this work is the first to study the adaptive adversary model in packet scheduling, i.e., all known randomized algorithms were designed and analyzed assuming the weaker oblivious adversary model. All upper bounds against adaptive adversaries were achieved by deterministic algorithms, while the lower bounds were implied by lower bounds for oblivious adversaries. In particular, the currently best lower and upper bounds for arbitrary instances against an adaptive adversary were 1.25 [6] and 1.828 [4], respectively.

We improve these bounds by presenting adaptively created 2-bounded instances which force any randomized algorithm to have a competitive ratio at least 4/3. We then extend our technique to obtain two more lower bounds: 4/3 for memoryless scale-invariant algorithms on 2-uniform instances and 1.2 for general algorithms on 2-uniform instances. We present an optimal 4/3-competitive algorithm Rand for 2-bounded instances; this algorithm is memoryless scale-invariant, so it matches two of our lower bounds. Finally, we give a new analysis of the memoryless scale-invariant algorithm RMix by Chin et al. [8], proving its 1.582-competitiveness against an adaptive adversary. We note that the original analysis essentially depended on the adversary's obliviousness, and thus did not extend to the adaptive adversary model.

### *1.3. Preliminaries*

For any algorithm $A$, let $\mathcal{B}_A$ denote the state of its buffer, that is, the set of packets stored in the buffer. In particular, $\mathcal{B}_{\text{Adv}}$ denotes the state of the buffer of the adversary Adv. We denote the expected gain of algorithm $A$ by $\mathcal{G}_A$.

We assume that time is slotted in the following way. Both Adv and the algorithm choose, independently, a packet and transmit it during a *step*, where each step $t$ corresponds to the interval $(t, t+1)$. Right after the algorithm (Adv) transmits a packet at a step $t$, this packet is immediately removed from its buffer. Then (at time $t+1$), the relative deadlines of all remaining packets are decremented by 1, and the packets whose relative deadlines reach 0 expire and are removed from both Adv's and the algorithm's buffers. Next, the adversary injects any set of packets, and their relative deadlines are initialized. At this point, we proceed to step $t+1$.

We denote a packet with weight $w$ and relative deadline $d$ by $(w, d)$. Upon injection, $d \in \{1, 2\}$ for 2-bounded instances, whereas $d = 2$ for 2-uniform ones. Thus, whenever we analyze such instances, we call packets with relative deadline 1 *tight*, and those with relative deadline 2 *loose*.

To no surprise, all known algorithms are *scale-invariant*, which means that they make the same decisions if all the weights of packets in an instance are scaled by a positive constant. A class of further restricted algorithms is of special interest for their simplicity. Informally, an algorithm is memoryless if in every step its decision depends only on the set of packets pending at $t$.

Formally, an online scheduling algorithm Alg is called *memoryless scale-invariant* if

1. the distribution of packet transmission probability depends only on the set of packets pending for Alg,
2. the distribution is invariant under scaling, i.e., multiplying the weights of the pending jobs by the same positive constant does not affect it.

## 2. Lower bounds

In this section, we show three lower bounds for the competitive ratio: (i) $\frac{4}{3}$ for algorithms on 2-bounded instances, (ii) $\frac{6}{5}$ for unrestricted algorithms on 2-uniform instances, and (iii) $\frac{4}{3}$ for memoryless scale-invariant algorithms on 2-uniform instances. To prove these bounds, we give finite strategies of the adversary that force the gain of any randomized algorithm to be at most $1/R + \epsilon$ times the gain of the adversary, where $R$ is the desired lower bound. As after executing this strategy the buffers of both the algorithm and the adversary are empty, we may repeat it, achieving arbitrarily high gain of the adversary. This, together with the fact that the $\epsilon$ term can be made arbitrarily small, implies the lower bound $R$ on the ratio of any randomized algorithm.

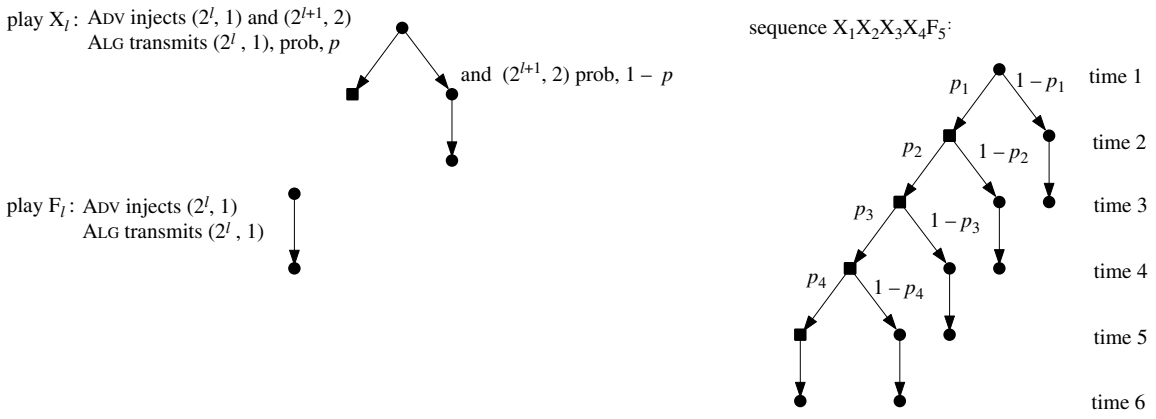


**Fig. 1.** Plays for 2-bounded instances and an example of a sequence of plays. Dots denote times, squares represent times which are continuation points. Arrows represent time steps and possible actions of the algorithm.

Without loss of generality, we assume that Alg always sends a packet if its buffer is non-empty.

### *2.1. 2-bounded sequences and unrestricted algorithms*

Instead of describing the whole strategy of the adversary at once, we partition it into building blocks called *plays*. Each play specifies the decisions of the adversary, namely packet injections and transmissions, in response to the algorithm's choices. Such description allows us to encapsulate all the repeatable details of the construction in single entities. We say that the *game ends* if the buffers of both algorithm and the adversary are empty; after this point the adversary will not inject any packets.

Throughout this section, we refer to the algorithm and the adversary as Alg and Adv, respectively.

*Plays* $\mathtt{X}_\ell$ *and* $\mathtt{F}_\ell$. We start by describing two plays. The first play is the most ubiquitous one: $\mathtt{X}_\ell$, parametrized by value $\ell$. Assume that at a certain time $t$, Alg's and Adv's buffers are either empty or contain a single tight packet $(2^\ell, 1)$. Then the adversary may use the strategy described by play $\mathtt{X}_\ell$ (we simply write "use $\mathtt{X}_\ell$"), which means the following:

- At time $t$, the adversary injects two packets, $(2^\ell, 1)$ and $(2^{\ell+1}, 2)$. In effect, both Alg and Adv now have one loose packet $(2^{\ell+1}, 2)$ and at least one copy of a tight packet $(2^\ell, 1)$.
- Further, let $p$ be the probability that Alg transmits the tight packet, $(2^\ell, 1)$, in step $t$. The adversary tries to transmit the packet different from the one transmitted by the algorithm, i.e., Adv transmits $(2^{\ell+1}, 2)$ if $p > 1/2$ and $(2^\ell, 1)$ otherwise. If Alg transmits $(2^{\ell+1}, 2)$, then its buffer becomes empty at time $t + 1$. In this case, in step $t + 1$, Adv transmits $(2^{\ell+1}, 1)$ if it still has it, and the game ends here. If Alg transmits $(2^\ell, 1)$, then it has $(2^{\ell+1}, 1)$ in its buffer at time $t + 1$.

With the same assumptions on the contents of the buffers as above (i.e., when $\mathcal{B}_{\text{Alg}}, \mathcal{B}_{\text{Adv}} \subseteq \{(2^\ell, 1)\}$), the adversary may also use a play $\mathtt{F}_\ell$, which simply means that the adversary injects $(2^\ell, 1)$. In the subsequent step, both Alg and Adv transmit $(2^\ell, 1)$ and the game ends. Observe that on play $\mathtt{F}_\ell$, the algorithm does not make any actual choice, i.e., its behavior is fixed.

Both plays are depicted in Fig. 1. Here, nodes denote (integer) times when the adversary injects some packets and edges denote the possible actions of the algorithm during the time step, i.e., which packet it chooses to transmit and with what probability.

*Combining plays into strategies.* The strategies of the adversary considered in our proofs will always be sequence of plays; in particular, the strategies used in this subsection have the form of $\mathtt{X}_1\mathtt{X}_2\mathtt{X}_3 \ldots \mathtt{X}_{n-2}\mathtt{X}_{n-1}\mathtt{F}_n$, where $n$ is fixed by the adversary and unknown to the algorithm. This strategy is employed in the following manner. At the beginning, the adversary uses play $\mathtt{X}_1$. Note that such action is feasible, because Alg and Adv start with empty buffers. Further, during play $\mathtt{X}_1$, the game either ends or it reaches a specific point (called *continuation point*), marked by square in Fig. 1. In the latter case, the properties of play $\mathtt{X}_1$ guarantee that $\mathcal{B}_{\text{Alg}} = \{(2^2, 1)\}$ and $\mathcal{B}_{\text{Adv}} \subseteq \{(2^2, 1)\}$, and thus Adv may use $\mathtt{X}_2$. This reasoning extends up to the continuation point of $\mathtt{X}_{n-1}$, where the adversary may finally use play $\mathtt{F}_n$. After that point, the game ends unconditionally in one step.

Let us recapitulate the description above. For each play, there are certain preconditions concerning the buffers of Alg and Adv which have to be met for the adversary to use this play. Further, there are certain postconditions that are fulfilled at the continuation point of the play.

**Observation 2.1.** *The preconditions for both* $\mathtt{X}_\ell$ *and* $\mathtt{F}_\ell$ *are* $\mathcal{B}_{\text{Alg}}, \mathcal{B}_{\text{Adv}} \subseteq \{(2^\ell, 1)\}$. *The postcondition for* $\mathtt{X}_\ell$ *is* $\mathcal{B}_{\text{Adv}} \subseteq \mathcal{B}_{\text{Alg}} = \{(2^{\ell+1}, 1)\}$. *There is no postcondition for* $\mathtt{F}_\ell$ *as it does not have a continuation point.*

Now, the sequence of plays is *valid* if for any two consecutive plays, the postcondition of the former implies the precondition of the latter. For example, the sequence $\mathtt{X}_3\mathtt{X}_4\mathtt{X}_5\mathtt{X}_6$ is valid. We say that a sequence is *game defining* if (i) it is valid, (ii) the precondition of the first play is met with empty buffers of ALG and ADV, and (iii) the last play does not have a continuation point. In particular, $\mathtt{X}_1\mathtt{X}_2\mathtt{X}_3 \ldots \mathtt{X}_{n-2}\mathtt{X}_{n-1}\mathtt{F}_n$ fulfills these conditions for any $n$. A game defining sequence of plays describes the whole strategy of the adversary, in which plays are used one by one. Alternatively, one may view a game defining sequence as a complete game tree in which all leaves correspond to the game endings (see an example in Fig. 1). Note that it may happen that due to the (random) choices of the algorithm, the game ends before it reaches some plays.

*Computing gains.* Let $T_n$ be the game defining sequence $\mathtt{X}_1\mathtt{X}_2\mathtt{X}_3 \ldots \mathtt{X}_{n-2}\mathtt{X}_{n-1}\mathtt{F}_n$. First, we observe that if we consider the behavior of ALG only on the inputs generated by an adversary that follows the strategies of $T_n$ (for different values of $n$), then the behavior of ALG is completely defined by an infinite sequence of probabilities $p_1, p_2, p_3, \ldots$, where $p_\ell$ is the probability that ALG transmits the tight packet within play $\mathtt{X}_\ell$, given that the game reaches $\mathtt{X}_\ell$. Observe that for $\ell < n$, $p_\ell$ is also the probability that the game reaches the continuation point in $\mathtt{X}_\ell$ in sequence $T_n$, provided that it reached the beginning of $\mathtt{X}_\ell$.

For a fixed ALG (i.e., for a fixed sequence of $\{p_\ell\}_{\ell=1}^{\infty}$) and for fixed $T_n$, by $\mathcal{G}_{\text{ALG}}(T_n)$ we denote the expected total gain of ALG when the adversary uses the strategy described by $T_n$. To make the calculations concise, we first compute gains on the respective plays: $\mathcal{G}_{\text{ALG}}(\mathtt{G})$ is the expected conditional gain achieved by ALG while the adversary is using play $\mathtt{G}$, given the event that game reached play $\mathtt{G}$. (To avoid clutter, we use the same notation $\mathcal{G}_{\text{ALG}}()$ for expected gains on games and plays; both notations are in fact compatible, since the first play in $T_n$ is reached with probability 1.) Thus, we obtain

$$\mathcal{G}_{\text{ALG}}(T_n) = \sum_{\ell=1}^{n-1} \left( \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \cdot \prod_{j=1}^{\ell-1} p_j \right) + \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot \prod_{j=1}^{n-1} p_j.$$

The important property of gains is that $\mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell)$ is well defined (and its value is the same) for all sequences $T_n$ where this play appears (i.e., for $n > \ell$). The same holds for $\mathcal{G}_{\text{ALG}}(\mathtt{F}_\ell)$ as it appears only in sequence $T_\ell$. Thus, we may compute the gains on plays for fixed $\{p_\ell\}_{\ell=1}^{\infty}$, without taking into account the actual choice of $n$. We define gains of ADV analogously.

The idea of the lower bound is as follows. We show that for any algorithm ALG and any $\epsilon > 0$, there exists $n$, such that $\mathcal{G}_{\text{ADV}}(T_n) \geq (\frac{4}{3} - \epsilon) \cdot \mathcal{G}_{\text{ALG}}(T_n)$. To this end, we relate the gains of any algorithm ALG and the adversary ADV on plays $\mathtt{X}_\ell$ and $\mathtt{F}_\ell$. In particular, we show that $\mathcal{G}_{\text{ADV}}(\mathtt{X}_\ell) \geq \frac{4}{3} \cdot \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell)$. However, the gains of ADV and ALG on $\mathtt{F}_n$ are the same. We circumvent this difficulty by showing that if $\mathtt{F}_n$'s contribution in the overall gain cannot be neglected, then the ratio between total gains of ADV and ALG on all $\mathtt{X}_\ell$-s is strictly larger than $4/3$.

*Relating gains on plays.* We note that the general construction for the 2-uniform variant (especially for unrestricted algorithms) is essentially the same, although the plays themselves are different. Thus, to avoid repetitions in the proof, we make our reasoning slightly more general than explicitly needed for this section. In particular, we define two constants: $R_{\text{b}} = 4/3$ and $C_{\text{b}} = 1/3$ and we use them till the end of this section.

**Lemma 2.2.** *Fix any algorithm* ALG *and the corresponding sequence of probabilities* $\{p_\ell\}_{\ell=1}^{\infty}$. *Then the following relations hold for any* $\ell$*:*

(i) $\mathcal{G}_{\text{ADV}}(\mathtt{X}_\ell) \geq R_{\text{b}} \cdot \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) + \max\{0, C_{\text{b}} \cdot (2p_\ell - 1) \cdot 2^\ell\}$,
(ii) $\mathcal{G}_{\text{ALG}}(\mathtt{F}_\ell) = \mathcal{G}_{\text{ADV}}(\mathtt{F}_\ell) = \frac{C_{\text{b}}}{R_{\text{b}}-1} \cdot 2^\ell$,
(iii) $\mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \geq \mathcal{G}_{\text{ALG}}(\mathtt{F}_\ell)$,

*where* $R_{\text{b}} = 4/3$ *and* $C_{\text{b}} = 1/3$.

**Proof.** Suppose that play $\mathtt{X}_\ell$ has been reached (at some time $t$) during the game. Thus, by the definition of $\mathtt{X}_\ell$, both ALG and ADV have $(2^\ell, 1)$ and $(2^{\ell+1}, 2)$ in their buffers at time $t$. In step $t$, ALG transmits packet $(2^\ell, 1)$ with probability $p_\ell$ and $(2^{\ell+1}, 2)$ with the remaining probability, gaining $p_\ell \cdot 2^\ell + (1-p_\ell) \cdot 2^{\ell+1}$. Note that the contribution of step $t+1$ to $\mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell)$ is zero: if ALG transmits packet $(2^\ell, 1)$, then game reaches the continuation point of the play, otherwise it transmits $(2^{\ell+1}, 2)$ and its buffer becomes empty. Thus,

$$\mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) = (2 - p_\ell) \cdot 2^\ell. \tag{1}$$

Now we compute the gain of ADV. If $p_\ell > 1/2$, then ADV transmits $(2^{\ell+1}, 2)$ in step $t$, and its gain in $\mathtt{X}_\ell$ is $2^{\ell+1}$. Otherwise, $p_\ell \leq 1/2$, and then ADV transmits $(2^\ell, 1)$ in step $t$. With probability $1-p_\ell$, the game does not reach the continuation point of $\mathtt{X}_\ell$ and the adversary may additionally transmit the remaining packet $(2^{\ell+1}, 1)$ in step $t+1$. Hence, in this case its expected gain on $\mathtt{X}_\ell$ is $2^\ell + (1 - p_\ell) \cdot 2^{\ell+1}$. By combining these two cases, we obtain

$$\mathcal{G}_{\text{ADV}}(\mathtt{X}_\ell) = \max\{2, 3 - 2p_\ell\} \cdot 2^\ell. \tag{2}$$

To derive (i), after substituting (1) and (2) into (i) and multiplying both sides by $3/2^\ell$, we reduce it to $\max\{6, 9 - 6p_\ell\} \geq 8 - 4p_\ell + \max\{0, 2p_\ell - 1\}$, which is easily seen to hold after subtracting $8 - 4p_\ell$ from both sides.

Further, we observe that

$$\mathcal{G}_{\text{ALG}}(\mathtt{F}_\ell) = \mathcal{G}_{\text{ADV}}(\mathtt{F}_\ell) = 2^\ell. \tag{3}$$

This immediately yields properties (ii) and (iii). □

*Gains on sequences $T_n$.* We now derive bounds relating the gains of ALG and ADV on complete sequences $T_n$.

**Lemma 2.3.** *Fix any algorithm* ALG *and the corresponding sequence of probabilities* $\{p_\ell\}_{\ell=1}^{\infty}$. *For any* $n \geq 1$, *it holds that* $\mathcal{G}_{\text{ADV}}(T_n) \geq R_b \cdot \mathcal{G}_{\text{ALG}}(T_n) - 2C_b$.

**Proof.** For any $1 \leq k < n$ define $T_n^k$ to be the suffix of $T_n$ starting at $X_k$, that is $T_n^k = \mathtt{X}_k\mathtt{X}_{k+1}\ldots\mathtt{X}_{n-1}\mathtt{F}_n$. For $k = n$, $T_n^n$ is simply the play $\mathtt{F}_n$. Let $\mathcal{G}_{\text{ALG}}(T_n^k)$ be the conditional gain on $T_n^k$, given that $\mathtt{X}_k$ is reached. We then have

$$\mathcal{G}_{\text{ALG}}(T_n^k) = \sum_{\ell=k}^{n-1} \left( \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \cdot \prod_{j=k}^{\ell-1} p_j \right) + \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot \prod_{j=k}^{n-1} p_j.$$

An analogous formula holds for $\mathcal{G}_{\text{ADV}}(T_n^k)$, the adversary conditional gain on $T_n^k$.

By a backward induction on $k$, we prove that for each $k$, it holds that

$$\mathcal{G}_{\text{ADV}}(T_n^k) \geq R_b \cdot \mathcal{G}_{\text{ALG}}(T_n^k) - C_b \cdot 2^k. \tag{4}$$

The induction base ($k = n$) is established by property (ii) of Lemma 2.2:

$$\mathcal{G}_{\text{ADV}}(T_n^n) = \mathcal{G}_{\text{ALG}}(T_n^n) = R_b \cdot \mathcal{G}_{\text{ALG}}(T_n^n) - (R_b - 1) \cdot \mathcal{G}_{\text{ALG}}(T_n^n) = R_b \cdot \mathcal{G}_{\text{ALG}}(T_n^n) - C_b \cdot 2^n.$$

In the inductive step, we assume that (4) holds for $k+1$ and we prove that it holds for $k$ as well. First, we observe that $\mathcal{G}_{\text{ALG}}(T_n^k) = \mathcal{G}_{\text{ALG}}(\mathtt{X}_k) + p_k \cdot \mathcal{G}_{\text{ALG}}(T_n^{k+1})$, and the same relation holds for $\mathcal{G}_{\text{ADV}}$. Then, by the inductive assumption (4),

$$\mathcal{G}_{\text{ADV}}(T_n^k) \geq \mathcal{G}_{\text{ADV}}(\mathtt{X}_k) + p_k \cdot \left( R_b \cdot \mathcal{G}_{\text{ALG}}(T_n^{k+1}) - C_b \cdot 2^{k+1} \right).$$

Now, by property (i) of Lemma 2.2,

$$\begin{aligned}
\mathcal{G}_{\text{ADV}}(T_n^k) &\geq \left( R_b \cdot \mathcal{G}_{\text{ALG}}(\mathtt{X}_k) + C_b \cdot 2^k \cdot \max\{0, 2p_k - 1\} \right) + \left( R_b \cdot p_k \cdot \mathcal{G}_{\text{ALG}}(T_n^{k+1}) - C_b \cdot p_k \cdot 2^{k+1} \right) \\
&\geq R_b \cdot \left( \mathcal{G}_{\text{ALG}}(\mathtt{X}_k) + p_k \cdot \mathcal{G}_{\text{ALG}}(T_n^{k+1}) \right) + C_b \cdot 2^k \cdot \max\{-2p_k, -1\} \\
&\geq R_b \cdot \left( \mathcal{G}_{\text{ALG}}(\mathtt{X}_k) + p_k \cdot \mathcal{G}_{\text{ALG}}(T_n^{k+1}) \right) - C_b \cdot 2^k \\
&= R_b \cdot \mathcal{G}_{\text{ALG}}(T_n^k) - C_b \cdot 2^k.
\end{aligned}$$

Finally, observe that the lemma follows from (4) for $k = 1$. □

Lemma 2.3 appears to imply our lower bound, but in reality is not sufficient, since we still need to deal with a possibility that the adversary gain for the whole game is small (constant). The theorem below wraps up the whole lower bound proof.

**Theorem 2.4.** *For* 2*-bounded instances, the competitive ratio of any randomized algorithm* ALG *against an adaptive adversary is at least* $R_b = 4/3$.

**Proof.** Let $\{p_\ell\}_{\ell=1}^{\infty}$ be the sequence of probabilities corresponding to ALG. Let $P_\ell$ be the probability of reaching the continuation point of the $\ell$-th play of $T_n$, i.e., $P_\ell = \prod_{i=1}^{\ell} p_i$, and let $P_0 = 1$. First, for any $n \geq 1$, we compare the gains of ALG on $T_{n-1}$ and on $T_n$:

$$\begin{aligned}
\mathcal{G}_{\text{ALG}}(T_n) &= \sum_{\ell=1}^{n-2} \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \cdot P_{\ell-1} + \mathcal{G}_{\text{ALG}}(\mathtt{X}_{n-1}) \cdot P_{n-2} + \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot P_{n-1} \\
&\geq \sum_{\ell=1}^{n-2} \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \cdot P_{\ell-1} + \mathcal{G}_{\text{ALG}}(\mathtt{F}_{n-1}) \cdot P_{n-2} + \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot P_{n-1} \\
&= \mathcal{G}_{\text{ALG}}(T_{n-1}) + \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot P_{n-1},
\end{aligned}$$

where in the inequality we used property (iii) of Lemma 2.2.

Thus, $\mathcal{G}_{\text{ALG}}(T_n)$ is a non-decreasing sequence of reals, and hence the sequence $\mathcal{G}_{\text{ALG}}(T_n)$ either diverges to infinity or converges to its supremum $g$. If $\lim_{n\to\infty} \mathcal{G}_{\text{ALG}}(T_n) = \infty$, then, by Lemma 2.3, it holds that $\limsup_{n\to\infty} \mathcal{G}_{\text{ADV}}(T_n)/\mathcal{G}_{\text{ALG}}(T_n) \geq R_b$, as desired.

So assume that $\lim_{n\to\infty} \mathcal{G}_{\text{ALG}}(T_n) = g < \infty$. In this case, the expected non-conditional gain on $\mathtt{F}_n$ becomes negligible, i.e., by the reasoning above, $\lim_{n\to\infty} \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot P_{n-1} = 0$. Then, expressing the gain on $T_n$ as a weighted sum of gains on individual plays and using property (i) of Lemma 2.2, we obtain

$$\begin{aligned}
\limsup_{n\to\infty} \frac{\mathcal{G}_{\text{ADV}}(T_n)}{\mathcal{G}_{\text{ALG}}(T_n)} &= \limsup_{n\to\infty} \frac{\sum_{\ell=1}^{n-1} \mathcal{G}_{\text{ADV}}(\mathtt{X}_\ell) \cdot P_{\ell-1} + \mathcal{G}_{\text{ADV}}(\mathtt{F}_n) \cdot P_{n-1}}{\sum_{\ell=1}^{n-1} \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \cdot P_{\ell-1} + \mathcal{G}_{\text{ALG}}(\mathtt{F}_n) \cdot P_{n-1}} \\
&= \limsup_{n\to\infty} \frac{\sum_{\ell=1}^{n-1} \mathcal{G}_{\text{ADV}}(\mathtt{X}_\ell) \cdot P_{\ell-1}}{\sum_{\ell=1}^{n-1} \mathcal{G}_{\text{ALG}}(\mathtt{X}_\ell) \cdot P_{\ell-1}} \\
&\geq R_b.
\end{aligned}$$

Therefore, we have shown that $\limsup_{n\to\infty} \frac{\mathcal{G}_{\text{ADV}}(T_n)}{\mathcal{G}_{\text{ALG}}(T_n)} \geq R_b$. This means that by taking large enough $n$ we can force the competitive ratio to be arbitrarily close to $R_b$. Moreover, the adversary may repeat the construction arbitrarily many times, to ensure that the gain is unbounded and cannot be absorbed by the additive constant of the competitive ratio. □

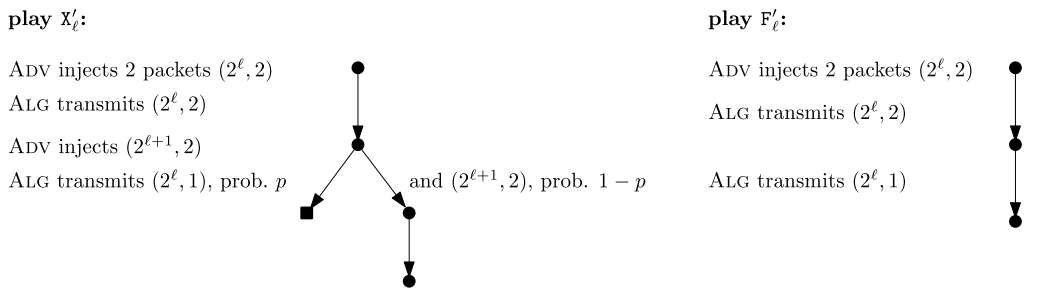


**Fig. 2.** Plays for 2-uniform instances.

### 2.2. 2-uniform sequences and unrestricted algorithms

In this section, we adapt the construction from the previous section to obtain a lower bound on the competitive ratio on 2-uniform instances. Note that the strategy of the adversary has to be changed, because injecting tight packets is no longer possible.

To this end, we create a play $\mathtt{X}'_\ell$ out of $\mathtt{X}_\ell$ in the following way. First, we remove the initial adversarial injection of the tight packet $(2^\ell, 1)$, as this is not feasible in the 2-uniform setting. Instead, we prepend an additional step at the beginning of which the adversary injects two loose packets $(2^\ell, 2)$. Note that either player may already have a tight packet $(2^\ell, 1)$ pending at the beginning of this step. But, in any case, we can assume that this tight packet is discarded, since only two of these three pending packets can be transmitted anyway.

The resulting play $\mathtt{X}'_\ell$ is depicted in Fig. 2. We also have to alter play $\mathtt{F}_\ell$ using the same transformation. In result, play $\mathtt{F}'_\ell$ consists of two steps and at its beginning the adversary injects two packets $(2^\ell, 2)$. Note that the preconditions and postconditions of $\mathtt{X}'_\ell$ and $\mathtt{F}'_\ell$ are the same as that of $\mathtt{X}_\ell$ and $\mathtt{F}_\ell$, respectively.

**Observation 2.5.** *The preconditions for both $\mathtt{X}'_\ell$ and $\mathtt{F}'_\ell$ are $\mathcal{B}_{\text{Alg}}, \mathcal{B}_{\text{Adv}} \subseteq \{(2^\ell, 1)\}$. The postcondition for $\mathtt{X}_\ell$ is $\mathcal{B}_{\text{Adv}} \subseteq \mathcal{B}_{\text{Alg}} = \{(2^{\ell+1}, 1)\}$. There is no postcondition for $\mathtt{F}'_\ell$ as it does not have a continuation point.*

By Observation 2.5, the sequence $T'_n = \mathtt{X}'_1\mathtt{X}'_2\mathtt{X}'_3 \dots \mathtt{X}'_{n-2}\mathtt{X}'_{n-1}\mathtt{F}'_n$ is game defining for any $n \geq 1$. We use the same approach as for 2-bounded sequences, but we replace plays $\mathtt{X}_\ell$ and $\mathtt{F}_\ell$ with $\mathtt{X}'_\ell$ and $\mathtt{F}'_\ell$, respectively. In particular, we run any algorithm Alg on sequences $T'_n$ and we identify Alg with infinite sequence of probabilities $\{p_\ell\}_{\ell=1}^\infty$, where $p_\ell$ is the probability of Alg transmitting the tight packet in the second step of play $\mathtt{X}'_\ell$.

First, we show a counterpart of Lemma 2.2 for plays $\mathtt{X}'_\ell$ and $\mathtt{F}'_\ell$ using different constants, namely $R_{\text{u}} = 6/5$ and $C_{\text{u}} = 2/5$.

**Lemma 2.6.** *Fix any algorithm Alg and the corresponding sequence of probabilities $\{p_\ell\}_{\ell=1}^\infty$. Then the following relations hold for any $\ell$:*

(i) $\mathcal{G}_{\text{Adv}}(\mathtt{X}'_\ell) \geq R_{\text{u}} \cdot \mathcal{G}_{\text{Alg}}(\mathtt{X}'_\ell) + \max\{0, C_{\text{u}} \cdot (2p_\ell - 1) \cdot 2^\ell\}$,
(ii) $\mathcal{G}_{\text{Alg}}(\mathtt{F}'_\ell) = \mathcal{G}_{\text{Adv}}(\mathtt{F}'_\ell) = \frac{C_{\text{u}}}{R_{\text{u}}-1} \cdot 2^\ell$,
(iii) $\mathcal{G}_{\text{Alg}}(\mathtt{X}'_\ell) \geq \mathcal{G}_{\text{Alg}}(\mathtt{F}'_\ell)$,

*where $R_{\text{u}} = 6/5$ and $C_{\text{u}} = 2/5$.*

**Proof.** During the first step of $\mathtt{X}'_\ell$ both Adv and Alg transmit packets of weight $2^\ell$. Afterward, their gain is the same as on $\mathtt{X}_\ell$. By Eqs. (1) and (2) in the proof of Lemma 2.2, this immediately yields $\mathcal{G}_{\text{Alg}}(\mathtt{X}'_\ell) = (3 - p_\ell) \cdot 2^\ell$ and $\mathcal{G}_{\text{Adv}}(\mathtt{X}'_\ell) = \max\{3, 4 - 2p_\ell\} \cdot 2^\ell$. Using an argument identical to that in the proof of Lemma 2.2, property (i) follows.

Within play $\mathtt{F}'_\ell$, both the algorithm and the adversary transmit two packets of value $2^\ell$, gaining $2^{\ell+1}$ in total. This proves properties (ii) and (iii). □

Now, we observe that proofs of Lemma 2.3 and Theorem 2.4 hold with virtually no changes. The only modification is replacing plays $\mathtt{X}_\ell$ and $\mathtt{F}_\ell$ by $\mathtt{X}'_\ell$ and $\mathtt{F}'_\ell$, respectively, and constants $R_{\text{b}}$ and $C_{\text{b}}$ by $R_{\text{u}}$ and $C_{\text{u}}$. Hence, we obtain the following corollary.

**Corollary 2.7.** *For 2-uniform instances, the competitive ratio of any randomized algorithm Alg against an adaptive adversary is at least $R_{\text{u}} = 6/5$.*

### 2.3. 2-uniform sequences and memoryless scale-invariant algorithms

In this section, we give a stronger lower bound on the competitive ratio of memoryless scale-invariant algorithms. The construction follows the pattern of our previous lower bounds. First, we split the play $\mathtt{X}'_\ell$ into two plays $\mathtt{I}_\ell$ and $\mathtt{K}_\ell$, where $\mathtt{I}_\ell$ is the first step (including the initial injection of two loose packets) of $\mathtt{X}'_\ell$ and $\mathtt{K}_\ell$ is the remaining part of this play. Then, the sequence used in the previous section is simply $T'_n = \mathtt{I}_1\mathtt{K}_1\mathtt{I}_2\mathtt{K}_2\mathtt{I}_3\mathtt{K}_3 \dots \mathtt{I}_{n-2}\mathtt{K}_{n-2}\mathtt{I}_{n-1}\mathtt{K}_{n-2}\mathtt{F}'_n$. By the construction of $\mathtt{I}_\ell$ and $\mathtt{K}_\ell$ and Observation 2.5, we may define preconditions and postconditions of these plays as follows.

**Observation 2.8.** *The precondition for both* $\mathtt{I}_\ell$ *and* $\mathtt{F}'_\ell$ *is* $\mathcal{B}_{\text{ALG}}, \mathcal{B}_{\text{ADV}} \subseteq \{(2^\ell, 1)\}$ *and the postcondition for* $\mathtt{I}_\ell$ *is* $\mathcal{B}_{\text{ALG}} = \mathcal{B}_{\text{ADV}} = \{(2^\ell, 1)\}$. *The precondition for* $\mathtt{K}_\ell$ *is* $\mathcal{B}_{\text{ALG}} = \mathcal{B}_{\text{ADV}} = \{(2^\ell, 1)\}$ *and its postcondition is* $\mathcal{B}_{\text{ADV}} \subseteq \mathcal{B}_{\text{ALG}} = \{(2^{\ell+1}, 1)\}$.

Note that the gains on $\mathtt{I}_\ell$ are the same for ADV and ALG, and thus these plays are the reason of a weaker lower bound for 2-uniform sequences. However, in the case of memoryless scale-invariant algorithms, we can make the incurred competitive ratio larger by removing all $\mathtt{I}_\ell$ except the first one from the sequence above. This means we analyze the behavior of an online algorithm on the sequence $T''_n := \mathtt{I}_1\mathtt{K}_1\mathtt{K}_2\mathtt{K}_3 \ldots \mathtt{K}_{n-2}\mathtt{K}_{n-1}\mathtt{F}'_n$. Note that (as in the previous constructions) for such sequences the behavior of ALG is completely defined by an infinite sequence of probabilities $p_1, p_2, p_3, \ldots$. This time, $p_\ell$ is the probability that ALG transmits the tight packet within play $\mathtt{K}_\ell$ given that the game reaches $\mathtt{K}_\ell$. Since we consider only memoryless scale-invariant algorithms, the probabilities $p_1, p_2, \ldots, p_{n-1}$ coincide, and they are denoted by $p$. We call $p$ the *probability used* by ALG.

First, we show that the adversary can indeed use the strategy described by $T''_n$, i.e., that it is game defining. We note that this is not true for arbitrary algorithms as at the end of $\mathtt{K}_\ell$ the buffer of ADV might be empty, whereas at the beginning of $\mathtt{K}_{\ell+1}$, ADV might want to transmit a packet $(2^{\ell+1}, 1)$.

**Lemma 2.9.** *For any n, the sequence* $T''_n = \mathtt{I}_1\mathtt{K}_1\mathtt{K}_2\mathtt{K}_3 \ldots \mathtt{K}_{n-2}\mathtt{K}_{n-1}\mathtt{F}'_n$ *is game defining for a memoryless scale-invariant algorithm* ALG.

**Proof.** By Observations 2.5 and 2.8, the preconditions of $\mathtt{I}_1$ are met with empty buffers of ALG and ADV and $\mathtt{F}'_n$ does not have a continuation point.

Thus, we only have to show that $T''_n$ is valid, i.e., that for any two consecutive plays of $T''_n$, the postcondition of the former implies the precondition of the latter. Again, by Observation 2.8, the only relation that does not follow immediately is that for $1 \le \ell \le n-2$, the postcondition of $\mathtt{K}_\ell$ concerning the contents of the buffer of ADV does not imply the precondition of $\mathtt{K}_{\ell+1}$. In the following, we take a closer look at this issue for different strategies of ALG. We consider two cases concerning the probability $p$ used by ALG.

Case 1: If $p \le 1/2$, then we may guarantee a stronger postcondition of $\mathtt{K}_\ell$, i.e., $\mathcal{B}_{\text{ADV}} = \{(2^{\ell+1}, 1)\}$. By the precondition, $\mathcal{B}_{\text{ADV}} = \{(2^\ell, 1)\}$, and at the beginning of $\mathtt{K}_\ell$ a loose packet $(2^{\ell+1}, 2)$ is injected. Since $p \le 1/2$, ADV transmits a tight packet $(2^\ell, 1)$ in the first step of $\mathtt{K}_\ell$. Hence, at the continuation point of the play $\mathtt{K}_\ell$ (reached with probability $p$), ADV still has packet $(2^{\ell+1}, 1)$ in the buffer.

Case 2: If $p > 1/2$, then the strategy of ADV works within $\mathtt{K}_\ell$ even if the precondition is not fulfilled, i.e., even if it does not have tight packet $(2^\ell, 1)$ in the buffer. At the beginning of $\mathtt{K}_\ell$, a loose packet $(2^{\ell+1}, 2)$ is injected. Since $p > 1/2$, ADV always chooses to transmit this loose packet, so the lack of the tight packet does not affect it.

Hence, $T''_n$ is valid for any choice of $p$. □

Now, we can compute the gains on particular plays of $T''_n$. Clearly, $\mathcal{G}_{\text{ALG}}(\mathtt{I}_1) = \mathcal{G}_{\text{ADV}}(\mathtt{I}_1) = 2$. Now, for any $\ell$, the gain on $\mathtt{K}_\ell$ is the same as gain on $\mathtt{X}_\ell$, and thus by Eqs. (1) and (2) in the proof of Lemma 2.2, $\mathcal{G}_{\text{ALG}}(\mathtt{K}_\ell) = (2-p)\cdot 2^\ell$ and $\mathcal{G}_{\text{ADV}}(\mathtt{K}_\ell) = \max\{2, 3-2p\}\cdot 2^\ell$. By Lemma 2.6, the gains on $\mathtt{F}'_\ell$ are $\mathcal{G}_{\text{ALG}}(\mathtt{F}'_\ell) = \mathcal{G}_{\text{ADV}}(\mathtt{F}'_\ell) = 2^{\ell+1}$. Thus, the following counterpart of the first two properties of Lemmas 2.2 and 2.6 is immediate.

**Lemma 2.10.** *Fix any memoryless scale-invariant algorithm* ALG, *i.e., choose its probability p. Then, the following relations hold for any* $\ell$*:*

(i) $\mathcal{G}_{\text{ADV}}(\mathtt{K}_\ell) \ge R_{\text{m}} \cdot \mathcal{G}_{\text{ALG}}(\mathtt{K}_\ell) + \max\{0, C_{\text{m}} \cdot (2p-1)\cdot 2^\ell\}$,
(ii) $\mathcal{G}_{\text{ALG}}(\mathtt{F}'_\ell) = \mathcal{G}_{\text{ADV}}(\mathtt{F}'_\ell) = \frac{C_{\text{m}}}{R_{\text{m}}-1}\cdot 2^\ell$,

*where* $R_{\text{m}} = 4/3$ *and* $C_{\text{m}} = 2/3$.

However, in contrast to the bound for unrestricted algorithms for 2-uniform sequences, we cannot directly apply Theorem 2.4 to prove the bound on the competitive ratio. One obstacle is that the relation $\mathcal{G}_{\text{ALG}}(\mathtt{K}_\ell) \ge \mathcal{G}_{\text{ALG}}(\mathtt{F}'_\ell)$ (which would be an analogue of property (iii) of Lemma 2.2) does not hold. But, more importantly, even if we could apply Theorem 2.4, it would not yield the desired bound, because it would only relate the gains of ALG and ADV on the sequence $\mathtt{K}_1\mathtt{K}_2\mathtt{K}_3 \ldots \mathtt{K}_{n-2}\mathtt{K}_{n-1}\mathtt{F}'_n$. However, on $T''_n$ both ALG and ADV have also an initial gain of 2 on play $\mathtt{I}_1$. Hence, we prove the lower bound in a more straightforward manner, using the fact that ALG uses the same probability in each step.

**Theorem 2.11.** *For 2-uniform instances, the competitive ratio of any randomized memoryless scale-invariant algorithm* ALG *against an adaptive adversary is at least* 4/3.

**Proof.** As mentioned above, we analyze the behavior of ALG on the game defining sequence $T''_n = \mathtt{I}_1\mathtt{K}_1\mathtt{K}_2\mathtt{K}_3 \ldots \mathtt{K}_{n-2}\mathtt{K}_{n-1}\mathtt{F}'_n$, where $n$ is a constant which will be fixed later by the adversary. Let $p$ denote the probability used by ALG. Note that the probability of reaching the play $\mathtt{K}_\ell$ is equal to $p^{\ell-1}$ and that of reaching the play $\mathtt{F}'_n$ is $p^{n-1}$. By splitting the gain on $T''_n$ into gains on particular plays, we obtain the formula

$$\mathcal{G}_{\text{ALG}}(T''_n) = \mathcal{G}_{\text{ALG}}(\mathtt{I}_1) + \sum_{\ell=1}^{n-1} \mathcal{G}_{\text{ALG}}(\mathtt{K}_\ell)\cdot p^{\ell-1} + \mathcal{G}_{\text{ALG}}(\mathtt{F}'_n)\cdot p^{n-1}. \tag{5}$$

An analogous relation holds for $\mathcal{G}_{\text{ADV}}$. As in the proof of Theorem 2.4, it is sufficient to show that $\limsup_{n\to\infty} \frac{\mathcal{G}_{\text{ADV}}(T''_n)}{\mathcal{G}_{\text{ALG}}(T''_n)} \geq \frac{4}{3}$. When $p < 1/2$, we may show it by a direct computation:

$$\lim_{n\to\infty} \frac{\mathcal{G}_{\text{ADV}}(T''_n)}{\mathcal{G}_{\text{ALG}}(T''_n)} = \lim_{n\to\infty} \frac{2 + \sum_{\ell=1}^{n-1} (3-2p)\cdot 2^\ell p^{\ell-1} + 2^{n+1}p^{n-1}}{2 + \sum_{\ell=1}^{n-1} (2-p)\cdot 2^\ell p^{\ell-1} + 2^{n+1}p^{n-1}} = \frac{2 + \frac{2\cdot(3-2p)}{1-2p}}{2 + \frac{2\cdot(2-p)}{1-2p}} = \frac{4}{3}.$$

Otherwise, $p \geq 1/2$. By properties (i) and (ii) of Lemma 2.10, the proof of Lemma 2.3 holds also for the sequence $\mathtt{K}_1\mathtt{K}_2\mathtt{K}_3 \ldots \mathtt{K}_{n-2}\mathtt{K}_{n-1}\mathtt{F}'_n$. We only have to replace $R_{\mathtt{b}}$ and $C_{\mathtt{b}}$ by $R_{\mathtt{m}}$ and $C_{\mathtt{m}}$, respectively. This yields the following relation for any $n \geq 1$:

$$\mathcal{G}_{\text{ADV}}(T''_n) - \mathcal{G}_{\text{ADV}}(\mathtt{I}_1) \geq R_{\mathtt{m}} \cdot \left(\mathcal{G}_{\text{ALG}}(T''_n) - \mathcal{G}_{\text{ALG}}(\mathtt{I}_1)\right) - 2C_{\mathtt{m}},$$

and thus $\mathcal{G}_{\text{ADV}}(T''_n) \geq (4/3)\cdot\left(\mathcal{G}_{\text{ALG}}(T''_n) - 2\right) - 2\cdot(2/3) + 2 = (4/3)\cdot\mathcal{G}_{\text{ALG}}(T''_n) - 2$. As $p \geq 1/2$, it holds that $\mathcal{G}_{\text{ALG}}(K_\ell)\cdot p^{\ell-1} \geq 2$, and hence, by (5), the gain of ALG is arbitrarily large if we take large enough $n$. Therefore, in the limit the additive constant $-2$ can be ignored, which yields $\limsup_{n\to\infty} \frac{\mathcal{G}_{\text{ADV}}(T''_n)}{\mathcal{G}_{\text{ALG}}(T''_n)} \geq \frac{4}{3}$. □

## 3. Upper bounds

In this section, we present two online randomized algorithms, both of which are memoryless scale-invariant. The first one, RAND, is designed for 2-bounded instances, on which it attains the competitive ratio of 4/3, matching the lower bounds from Sections 2.1 and 2.3.

The other algorithm, RMIX, applies to arbitrary instances and is $\mathrm{e}/(\mathrm{e}-1)$-competitive. This algorithm was proposed by Chin et al. [8] and shown to be $\mathrm{e}/(\mathrm{e}-1)$-competitive in the oblivious adversary model. The analysis presented in [8] does not apply to the adaptive adversary case, for the following reasons: one, it employs a potential function that depends on the fixed adversary's schedule, and two, it assumes that the adversary follows an earliest-deadline-first schedule. The latter assumption is justified by the former, which in turn is justified by the fact that the instance is fixed in advance by the adversary. In the adaptive adversary model, however, both adversarial injections and transmissions depend on the random choices of the algorithm, hence the adversary's schedule is not fixed. Our proof technique, as described below, allows us to prove that RMIX remains $\mathrm{e}/(\mathrm{e}-1)$ against adaptive adversaries.

In our analyses, we follow the paradigm of modifying the adversary's buffer, introduced by Li et al. [14]. Namely, we assume that in each step the algorithm and the adversary have precisely the same pending packets in their buffers. Once they both transmit a packet, we modify the adversary's buffer judiciously to make it identical with that of the algorithm. Although, fundamentally, this is still amortized analysis (which is unavoidable in competitive analysis), this trick leads to a more streamlined and more intuitive proof that does not use any potential function.

When modifying the buffer, we may have to let the adversary transmit another packet and keep the one originally transmitted in the buffer, or upgrade one of the packets in its buffer by increasing its weight or deadline. We will ensure that these changes will be *advantageous to the adversary* in the following sense: for any adversary strategy ADV, starting from the current step and the buffer content, there is an adversary strategy $\overline{\text{ADV}}$ that continues computation with the modified buffer, such that the total gain of $\overline{\text{ADV}}$ starting (and including) the current step, on any instance, is at least as large as that of ADV.

To prove $R$-competitiveness, we show that in each step the expected *amortized gain* of the adversary is at most $R$ times the expected gain of the algorithm, where the former is the total weight of the packets that ADV eventually transmitted in this step. Both expected values are taken over random choices of the algorithm.

As mentioned before, we cannot assume that the adversary follows the earliest-deadline-first policy. The only assumption we make is that the adversary never transmits a packet $\mathtt{a}$ if there is another pending packet $\mathtt{b}$ such that transmitting $\mathtt{b}$ is always advantageous to the adversary. Formally, we introduce a dominance relation among the pending packets and assume that the adversary never transmits a dominated packet.

We say that a packet $\mathtt{a} = (w_a, d_a)$ is *dominated* by a packet $\mathtt{b} = (w_b, d_b)$ at time $t$ if at time $t$ both $\mathtt{a}$ and $\mathtt{b}$ are pending, $w_a \leq w_b$ and $d_a \geq d_b$. If one of these inequalities is strict, we say that $\mathtt{a}$ is *strictly dominated* by $\mathtt{b}$.

The following fact can be shown by the standard exchange argument.

**Fact 3.1.** *For any adversary strategy* ADV*, there is a strategy* $\overline{\text{ADV}}$ *with the following properties:*

(i) *the gain of* $\overline{\text{ADV}}$ *on every sequence is at least the gain of* ADV*,*
(ii) *in every step* $t$*,* $\overline{\text{ADV}}$ *does not transmit a strictly dominated packet at time* $t$*.*

### 3.1. Algorithm for 2-bounded instances

For simplicity, we assume that at least one tight and at least one loose packet is injected in each step, since otherwise 0-weight packets can be injected without changing the optimum solution. We describe the algorithm's behavior in a single step.

**Algorithm** RAND. Let $\mathsf{a} = (w_a, 1)$ and $\mathsf{b} = (w_b, 2)$ be the heaviest tight and loose pending packet, respectively. RAND transmits $\mathsf{a}$ with probability $\min\left\{\frac{w_a}{w_b}, 1\right\}$ and $\mathsf{b}$ with the remaining probability.

Note that, by Fact 3.1, ADV also transmits either $\mathsf{a}$ or $\mathsf{b}$, since exactly the same packets are pending for ADV, and every other packet is dominated by one of $\mathsf{a}$, $\mathsf{b}$. Specifically, $\mathsf{a}$ dominates all other tight packets and $\mathsf{b}$ dominates all other loose packets.

**Theorem 3.2.** RAND *is* $\frac{4}{3}$*-competitive against an adaptive adversary on* 2*-bounded instances.*

**Proof.** Note that if $w_a \geq w_b$ then RAND transmits $\mathsf{a}$. In this case $\mathsf{b}$ is dominated by $\mathsf{a}$, so ADV also transmits $\mathsf{a}$ by Fact 3.1. Thus ADV's buffer is the same as RAND's buffer after this step, and no further changes to it are needed. The ratio of the gains is 1.

In the rest of the proof we focus on the case $w_a < w_b$. In this case, the algorithm's expected gain in one step can be bounded as follows:

$$\mathcal{G}_{\text{RAND}} = \frac{w_a}{w_b} \cdot w_a + \left(1 - \frac{w_a}{w_b}\right) \cdot w_b = \frac{1}{w_b}\left(\left(w_a - \frac{1}{2}w_b\right)^2 + \frac{3}{4}w_b^2\right) \geq \frac{3}{4}w_b. \tag{6}$$

Now we describe the changes to ADV's scheduling policy and its buffer in the given step. We have two cases, depending on which packet is transmitted by ADV.

Case 1: ADV transmits $\mathsf{a}$. If RAND transmits $\mathsf{a}$ as well, which it does with probability $\frac{w_a}{w_b}$, we are done. If RAND transmits $\mathsf{b}$, we allow ADV to transmit both $\mathsf{a}$ and $\mathsf{b}$ in this step, which makes its buffer identical to RAND's afterward. The adversary's expected gain is

$$\mathcal{G}_{\text{ADV}} = \frac{w_a}{w_b} \cdot w_a + \left(1 - \frac{w_a}{w_b}\right) \cdot (w_a + w_b) = w_b.$$

Case 2: ADV transmits $\mathsf{b}$. If RAND transmits $\mathsf{b}$ as well, we are done. Otherwise, we allow ADV to keep a copy of $\mathsf{b}$ in its buffer. As $\mathsf{a}$ expires right away, the buffers of RAND and ADV are identical after this step. The adversary's gain is $w_b$ in this case.

These changes make ADV's and RAND's buffers identical and, furthermore, make the expected gain of the adversary equal exactly $w_b$. This, together with (6) yields the desired bound. □

### 3.2. Algorithm for unrestricted instances

We describe the algorithm's behavior in a single step. We assume that there are always some pending packets, for otherwise zero-weight packets can be issued.

**Algorithm** RMIX. Let $\mathsf{h} = (w_h, d_h)$ be the heaviest pending packet. Select a real number $x \in [-1, 0]$ uniformly at random. Transmit the earliest-deadline packet $\mathsf{f} = (w_f, d_f)$ such that $w_f \geq \mathrm{e}^x \cdot w_h$. (Break ties in favor of heavier packets.)

Notice that the algorithm never transmits a packet that is strictly dominated by another packet.

**Theorem 3.3.** RMIX *is* $\frac{\mathrm{e}}{\mathrm{e}-1}$*-competitive against an adaptive adversary.*

**Proof.** For a given step, we describe the changes to ADV's scheduling decisions and we modify its buffer to make it the same as RMIX's buffer. We then prove that in this step the ratio of ADV's amortized expected gain (that is, the gain of its modified strategy) to RMIX's expected gain is at most $\frac{\mathrm{e}}{\mathrm{e}-1}$. The theorem follows by summation over all steps.

Recall that $\mathsf{f} = (w_f, d_f)$ is the packet transmitted by RMIX. The weight $w_f$ of $\mathsf{f}$ is a random variable, and we will sometimes denote it by $w_{f(x)}$ when we want to emphasize its dependence on $x$. (Of course, the value of $d_f$ is a random variable as well,

but we will not be concerned with its distribution.) In this notation, the expected gain of RMix in a single step is

$$\mathcal{G}_{\text{RMIX}} = \mathbf{E}[w_{f(x)}] = \int_{-1}^{0} w_{f(x)}\, dx.$$

We now describe modifications to Adv's buffer and estimate Adv's amortized gain, for a fixed choice of $x \in [-1, 0]$. Assume that Adv transmits a packet $\mathsf{j} = (w_j, d_j)$. Without loss of generality, we assume that j is not strictly dominated by another packet (see Fact 3.1). We consider two cases.

Case 1: $d_f \le d_j$. Then $w_f \le w_j$, since j is undominated. After both Adv and RMix transmit their packets, we replace f in the buffer of Adv by a copy of j. This way their buffers remain the same afterward, and the change is advantageous to Adv: this is essentially an upgrade of the packet f in its buffer, as both $d_f \le d_j$ and $w_f \le w_j$ hold.

Case 2: $d_f > d_j$. After both Adv and RMix transmit their packets, we let Adv additionally transmit f and keep a copy of j in its buffer, which is clearly advantageous to Adv. This makes the buffers of Adv and RMix identical afterward.

Therefore Adv always gains $w_j$, and if $d_j < d_f$ he additionally gains $w_f$. Recall that neither j nor f is strictly dominated by another packet. Thus, by the choice of f, inequality $d_j < d_f$ is equivalent to $w_j < \mathrm{e}^x \cdot w_h$. This latter inequality is in turn equivalent to $x > \ln(w_j/w_h)$.

We are now ready to estimate Adv's amortized gain. Letting $y = \max\{\ln(w_j/w_h), -1\}$, from the discussion in the previous paragraph we have

$$\mathcal{G}_{\text{ADV}} = w_j + \mathbf{E}\left[w_{f(x)} | d_{f(x)} > d_j\right] = w_j + \int_{y}^{0} w_{f(x)}\, dx.$$

Finally, we compare the gains, obtaining

$$\frac{\mathcal{G}_{\text{ADV}}}{\mathcal{G}_{\text{RMIX}}} = \frac{w_j + \int_y^0 w_{f(x)}\, dx}{\int_{-1}^0 w_{f(x)}\, dx} = \frac{w_j + \int_y^0 w_{f(x)}\, dx}{\int_{-1}^y w_{f(x)}\, dx + \int_y^0 w_{f(x)}\, dx}.$$

Note that $\int_{-1}^{y} w_{f(x)}\, dx \le w_j$, as $y \le 0$ and $w_{f(x)} \le w_j$ for $x \le y$ (since j is undominated). Thus, the ratio is maximized when the value of $w_{f(x)}$ is minimized for all $x$. Hence,

$$\frac{\mathcal{G}_{\text{ADV}}}{\mathcal{G}_{\text{RMIX}}} \le \frac{w_j + \int_y^0 \mathrm{e}^x w_h\, dx}{\int_{-1}^0 \mathrm{e}^x w_h\, dx} \le \frac{w_j + w_h \cdot (1 - \frac{w_j}{w_h})}{w_h \cdot (1 - \frac{1}{\mathrm{e}})} = \frac{\mathrm{e}}{\mathrm{e} - 1},$$

which concludes the proof. □

## Acknowledgements

We would like to thank the anonymous referees for helpful comments and for pointing out some mistakes in the earlier version of this paper.

This work was supported by MNiSW grants number N N206 1723 33, 2007–2010 and N N206 490638, 2010–2011, and by NSF grants OISE-0340752 and CCF-0729071.

## References

[1] A. Kesselman, Z. Lotker, Y. Mansour, B. Patt-Shamir, B. Schieber, M. Sviridenko, Buffer overflow management in QoS switches, SIAM Journal on Computing 33 (2004) 563–583. Also appeared in Proc. of the 33rd Symposium on Theory of Computing, STOC'01, 2001, pp. 520–529.
[2] A. Borodin, R. El-Yaniv, Online Computation and Competitive Analysis, Cambridge University Press, 1998.
[3] S. Ben-David, A. Borodin, R.M. Karp, G. Tardos, A. Wigderson, On the power of randomization in online algorithms, Algorithmica 11 (1994) 2–14. Also appeared in Proc. of the 22nd Symposium on Theory of Computing, STOC'90, 1990, pp. 379–386.
[4] M. Englert, M. Westermann, Considering suppressed packets improves buffer management in QoS switches, in: Proc. of the 18th ACM-SIAM Symp. on Discrete Algorithms, SODA'07, SIAM, 2007, pp. 209–218.
[5] N. Andelman, Y. Mansour, A. Zhu, Competitive queueing policies for qos switches, in: Proc. of the 14th ACM-SIAM Symp. on Discrete Algorithms, SODA'03, SIAM, 2003, pp. 761–770.
[6] F.Y.L. Chin, S.P.Y. Fung, Online scheduling for partial job values: does timesharing or randomization help? Algorithmica 37 (2003) 149–164.
[7] B. Hajek, On the competitiveness of online scheduling of unit-length packets with hard deadlines in slotted time, in: Conference in Information Sciences and Systems, Johns Hopkins University, 2001, pp. 434–438.
[8] F.Y.L. Chin, M. Chrobak, S.P.Y. Fung, W. Jawor, J. Sgall, T. Tichý, Online competitive algorithms for maximizing weighted throughput of unit jobs, Journal of Discrete Algorithms 4 (2006) 255–276.
[9] M. Chrobak, W. Jawor, J. Sgall, T. Tichý, Improved online algorithms for buffer management in QoS switches, ACM Transactions on Algorithms 3 (2007). Also appeared in Proc. of the 12th European Symp. on Algorithms, ESA'04, 2004, pp. 204–215.
[10] A. Zhu, Analysis of queueing policies in QoS switches, Journal of Algorithms 53 (2004) 137–168.
[11] Y. Azar, Online packet switching, in: Proc. of 2nd Workshop on Approximation and Online Algorithms, WAOA'04, Springer, 2004, pp. 1–5.
[12] L. Epstein, R. van Stee, Buffer management problems, Sigact News 35 (2004) 58–66.
[13] M. Goldwasser, A survey of buffer management policies for packet switches, SIGACT News 41 (2010) 100–128.
[14] F. Li, J. Sethuraman, C. Stein, An optimal online algorithm for packet scheduling with agreeable deadlines, in: Proc. of the 16th ACM-SIAM Symp. on Discrete Algorithms, SODA'05, SIAM, 2005, pp. 801–802.